Kernel-Based Clustering: A Comparative Study

Man Singh[1], Sham Bansal[2], Ashish Kumar Garg[3], Mohammad Amir[4], Jarrar Ahmed[5] Suresh Kumar Yadav[6]

[1]University of Delhi, Department of Mathematics, SPM College (For Women), West Punjabi Bagh, New Delhi-110026

[2]University of Delhi, Department of Mathematics, Bharati College, Janakpuri, New Delhi-110058

[3]University of Delhi, Department of Mathematics, Janki Devi Memorial College, Rajender Nagar, Delhi-110060

[4]University of Delhi, Department of Mathematics, Indraprastha College for Women, Civil Lines, New Delhi-110054

[5]University of Delhi, Department of Mathematics, Dyal Singh College, Lodi Road, Delhi-110003

[6]University of Delhi, Department of Mathematics, Ramjas College, University Enclave, Delhi-110007

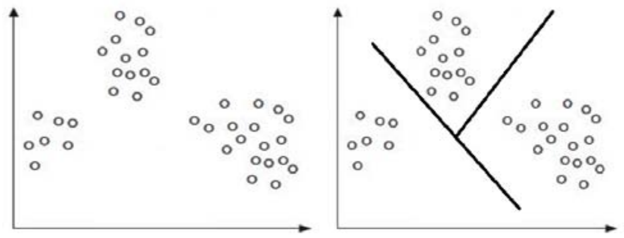
Abstract: We present a comprehensive comparative analysis of classical Fuzzy C-Means (FCM) clustering and kernel –based Fu*zzy* ***C-Means clustering. While Fuzzy C-Means is a popular soft-clustering method, its effectiveness is largely limited to hyper spherical clusters that are linearly separable. If the clusters are not linearly separable, the FCM is not particularly effective. By applying the kernel trick, the kernelised fuzzy c-means algorithm attempts to address this problem. The Kernel FCM operates by first non-linearly mapping the data to appropriate and sufficiently high-dimensional feature spaces, where according to the Mercer's theorem the data are likely to be linearly separable, and then applying the classical FCM algorithm. We first present the Hard C-Means clustering algorithm and a generalization of it called the Fuzzy C-Means. Then we explore the mathematical basis behind the kernel trick, both in general and especially in the setting of clustering. Following this we evaluate the performance gains provided by kernelised FCM and its classical counterpart, which is the main objective of the present work. It is shown that kernelised FCM does provide significant improvements for several popular Machine Learning data sets. However, it is observed that the performance of kerneized FCM depends greatly on the selection of the kernel parameters. We do a short comparative study of Kernelized FCM using different kernels.***

Keywords: Clustering, Fuzzy c – Means, Kernel Method, Mercer's Theorem, Kernel Trick.

1. Introduction

1.1 Clustering

Cluster analysis [1] divides data into groups (clusters) in order to improve our understanding of the data or to find interesting structures in data. Clustering algorithms divides up a data set into clusters or classes, where similar data objects are assigned to the cluster.



(a) Given data set (b) 3 well- separated clusters
Figure 1.1: 3 linearly separable clusters

In Figure (1.1) it is clear there are three clusters because of the spatial proximity between the data points .We can separate these clusters by a plane or a line and such data sets are called linearly separable. Now let us consider Figure (1.2) there is two circular clusters with different radii but in this case it is obvious that we cannot separate these two clusters by line or plane. The data points are not linearly separable in $\mathbf{R}^2$. Now the idea is to use a non- linear function to map the data from the low dimensional space $\mathbf{R}^2$ to some higher dimensional space $\mathbf{R}^p, p>2$ in such a way that the data are linearly separable $\mathbf{R}^p$.

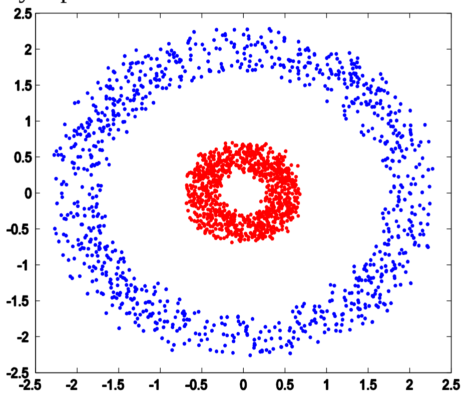


Figure (1.2): Description about the ring data points

1.2 The Kernel Trick

One interesting option available here is the Kernel trick. Let the data $\{\overline{x}_i\}$ come from q- dimensional Euclidean space $\mathbf{R}^p$.

The following forms the basis for the Kernel trick:

(i) There exists some higher dimensional feature space $H \subset \mathbf{R}^p$, usually $p >> q$ and a non-linear map: $\Phi : \mathbf{R}^p \rightarrow H \subset \mathbf{R}^p$ such that the transformed data points $\{\Phi(\overline{x}_i)\}$ .are linearly separable in H. This is guaranteed by Cover's theorem given below:

1.3 Cover's Theorem

A complex pattern classification Problem, cast in a high dimensional space nonlinearity, is more likely to be linearly separable than in a low- dimensional space, provided that the space is not densely populated.

(ii) The algorithm that needs to be kernelized depends only on the inner- product of the data points, i.e., only on the values of $\langle \overline{x_i}, \overline{x_j} \rangle$. For instance, the algorithms that make use of the Covariance matrices of the data where each entry is precisely the above inner product. The kernel trick now is in finding a scalar map $K : \mathbf{R}^p \times \mathbf{R}^q \to \mathbf{R}$ such that

$$k(\overline{x_i}, \overline{x_j}) = \langle \Phi(\overline{x_i}), \Phi(\overline{x_j}) \rangle,$$

i.e. the inner product of the transformed data points is the image of the original data points under K.

2. Fuzzy c- Means

In HCM, the membership degrees u_{ki} are either 0 or 1. What this means is that once a data point is assigned to a cluster center other cluster centers do not see it at all. This often leads to a much skewed partition. Towards avoiding this pitfall, the Fuzzy c- Means algorithm [2] proceeds to **fuzzify** the constraint μ_b. That is, we allow fuzzy membership by relaxing the condition $\mu_{ki} \in (0,1)$ into the fuzzy one; $\mu_{ki} \in [0,1]$. This means that we replace μ_b .by next constraint:

$$\mu_f = U = (u_{ki}) : \sum_{j=1}^{n} u_{kj} = 1, 1 \le k \le N;$$

$$\text{and } u_{ki} \in [0,1], 1 \le k \le N, 1 \le i \le c. \qquad (1)$$

2.1 Algorithm FCM

In the following we give the basic procedure of Fuzzy c- Means.

FCM1: [Generate initial value:] Generate c initial value for cluster centers $\overline{V} = (\overline{v_1}, \overline{v_2}, \overline{v_3}.........., \overline{v_n})$.

FCM2. [Find optimal U:] Calculate

$$\overline{U} = \arg\min_{U \in \mu_f} J_{fcm}(\mathrm{U}, \overline{V}). \qquad (2)$$

FCM3. [Find optimal V:] Calculate

$$\overline{V} = \arg\min_{V} J_{fcm}(\overline{U}, \mathrm{V}). \qquad (3)$$

FCM4. [Test convergence:] If $\overline{U}$ or $\overline{V}$ is convergent, stop; else go to FCM2.

End FCM.

3. Kernels and the Kernel Trick

Kernelization [2] is an efficient way to transform a data- set (usually low dimensional) to a higher dimensional data set. The Kernel Trick, as it is usually referred to, hinges on the Mercer's theorem which guarantees that there is mapping $\Phi(x) : \mathbf{R}^p \to H$, such that

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle \qquad (4)$$

Where H is Hilbert space.

3.1 Positive Definite Kernel

Let X is a non empty set. A function k on X × X which for all $m \in \mathrm{N}$ and all $x_1, x_2, ..., x_m \in X$ gives rise to a positive definite Gram matrix is called a positive definite kernel. A function k which gives rise to an operator T_k [4], [5], [7], via

$$(T_k f)(x) = \int_x k(x, y) f(y) dy \qquad (5)$$

is called the kernel of T_k.

Let $x, y \in \mathbf{R}^n$. In the following, we list some examples of commonly employed kernels:

- **The Polynomial Kernel :**

$k(x, y) = (\langle x, y \rangle + c)^d$ Where $c, d \in \mathbf{R}$ and d refers to the dimension of H.

- **The Gaussian Kernel :**

$$k(x, y) = \exp\left(\frac{-\lambda \|x - y\|^2}{2\sigma^2} \right) \qquad (6)$$

- **Tan- Hyperbolic Kernel :**

$$k(x, y) = 1 - \tanh\left(\frac{-\|x - y\|^2}{\sigma^2} \right) \qquad (7)$$

3.2 Mercer Theorem

Theorem: Suppose $k \in L_\infty(X^2)$ is a symmetric real-valued function such that the integral operator (1)

$T_K : L_2(X) \to L_2(X)$

$$(T_k f)(x) := \int_x k(x, y) f(y) d\mu(y) \qquad (8)$$

is positive definite; that is , for all $f \in L_2(X)$, we have

$$\int_x k(x, y) f(x) f(y) d\mu(x) d\mu(y) \ge 0. \qquad (9)$$

Let $\psi_j \in L_2(x)$ be the normalized orthogonal Eigen functions of T_k associated with the Eigen-values $\lambda_j > 0$, sorted in non-increasing order. Then

- $(\lambda_j) \in l_{1,}$
- $k(x, y) = \sum_{j=1}^{N_H} \lambda_j \varphi_j(x) \varphi_j(y)$ holds for almost all (x, y).

Mercer's [6], [7] result plays a central role in the kernel trick for the following reason. If k is a kernel satisfying the conditions of Mercer Theorem, one can constraint a map Φ into a space where k acts as a dot product,

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle, \qquad (10)$$

For almost all $x, y \in X$. Moreover given any $\in > 0$, there exist a map Φ_n into an n- dimensional dot product space such that

$$\left| k(x,y) - \langle \Phi^n(x), \Phi^n(y) \rangle \right| < \in \qquad (11)$$

for almost all $x, y \in X$. Both Mercer kernels and positive definite kernels can thus be represented as dot products in Hilbert spaces.

3.3 Kernelized fuzzy c- Means Algorithm

The method of fuzzy c- means uses the alternate optimization of

$$J_{fcm}(U,W) = \sum_{i=1}^{c} \sum_{k=1}^{N} (u_{ki})^m \left\| x_k - v_i \right\|^2 \qquad (12)$$

If we intend to analyze data using a kernel function, we should transform data into the high- dimensional feature space, in other words transformed objects $\Phi(x_1), \Phi(x_2), \ldots\ldots \Phi(x_N)$ should be divided into clusters.

The following objective functions [2] should therefore be considered

$$J_{kfcm}(U,W) = \sum_{i=1}^{c} \sum_{k=1}^{N} (u_{ki})^m \left\| \Phi(x_k) - W_i \right\|_H^2, \qquad (13)$$

Where $W = (\mathrm{W}_1, \mathrm{W}_2, \ldots\ldots, W_c)$ and W_i is the cluster centre in H; $\| \;\|_H$ is the form of H.

The alternate optimization algorithm FCM should to apply to (7). The optimal U is

$$\overline{u_{ki}} = \left[\sum_{j=1}^{c} \left(\frac{\left\| \Phi(x_k) - \overline{W_i} \right\|_H^2}{\left\| \Phi(x_k) - \overline{W_j} \right\|_H^2} \right)^{\frac{1}{m-1}} \right]^{-1}$$

$$= \left[\sum_{j=1}^{c} \left(\frac{D_{ki}}{D_{kj}} \right)^{\frac{1}{m-1}} \right]^{-1} \qquad (14)$$

For $J_{kfcm}(U,W)$. Notice that we put $\left\| \Phi(x_k) - \overline{W}_i \right\|_H^2$. The optimal W is given by

$$\overline{W_i} = \frac{\sum_{k=1}^{N} (\overline{u}_{ki})^m \, \Phi(x_k)}{\sum_{k=1}^{N} (\overline{u}_{ki})^m} \qquad (15)$$

For $J_{kfcm}(U,W)$.

$$\mathrm{D}_{ki} = \mathrm{K}_{kk} - \frac{2}{S_i(m)} \sum_{j=1}^{N} (\mathrm{u}_{ji})^m K_{jk} + \frac{1}{S_i(m)^2} \sum_{j=1}^{N} \sum_{b=1}^{N} (u_{ji}\, \mathrm{u}_{bi})^m \mathrm{K}_{jb}. \qquad (16)$$

Where $\mathrm{S}_i(m) = \sum_{k=1}^{N} (\overline{\mathrm{u}}_{ki})^m$ and

$$K_{kb} = k(x_k, x_b) = \langle \Phi(x_k), \Phi(x_b) \rangle.$$

Notice also that when the entropy method is used, the same equation (10) is used with m = 1.

Since we do not use the cluster centers in the kernalized method, the algorithm should be rewritten using solely U and D_{ki}. There is also a problem of how to determine the initial value of D_{ki}. For this purpose we select c objects $y_1, y_2, \ldots, y_c$ randomly from $(x_1, x_2, \ldots, x_N)$ and let $W_i = y_i (1 \le i \le c)$. Then

$$D_{ki} = \left\| \Phi(x_k) - \Phi(y_i) \right\|_H^2 = k(x_k, x_k) + k(y_i, y_i) - 2k(x_k, \mathrm{y}_i). \quad (17)$$

3.3.1 Algorithm of Kernelized FCM

KFCM1. Select randomly $y_1, y_2, \ldots, y_c \in (x_1, x_2, \ldots, x_N)$.

Calculate D_{ki} by (17).

KFCM2. Calculate u_{ki} by (14).

KFCM3. If the solution $U = (u_{ki})$ is convergent, stop. Else updates D_{ki} using (16). Go to KFCM2.

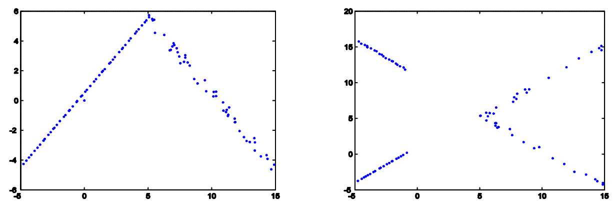
End KFCM.

4. Results and Discussion

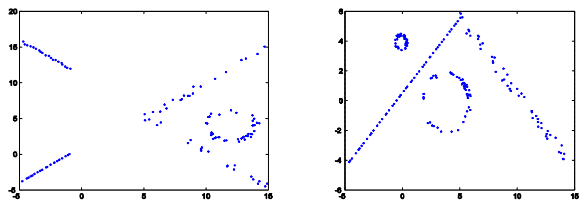
Some final concluding remarks detailing some further observations and possible future extensions are given.

4.1 The Data Sets

We consider the following 4 data sets some of which are linearly separable and the rest are not.

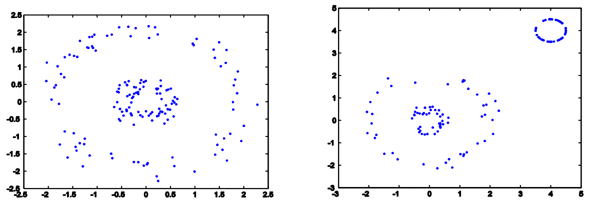


(a) Purely Linear - 1 or 2 Clusters (b) Purely Linear - 2,3 or 4 Clusters

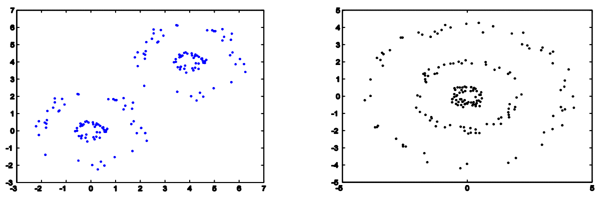


(c) Linear and 1 Non-linear clusters (d) Linear and 2 Non-linear clusters

Figure 4.1: Data sets that are both Purely Linear and Linear-cum-Non-linear.

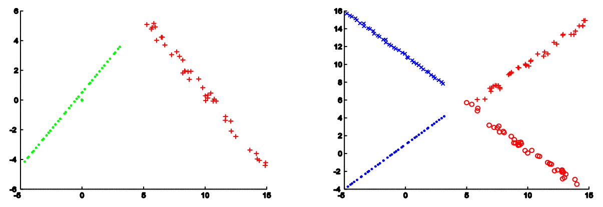


(a)Annular Ring (b) Annular ring and Circle


(c) Two Distinct Annular Rings (d) Three Annular Circles
Figure 4.2: Data sets that are purely Non-linear.

4.2 FCM on the Data Sets

We firstly apply the FCM algorithm on each of these. Figs. 4.3 and 4.4 contain the clusters that are usually found from repeated runs of the FCM on these data sets. It is clear that when the data are linearly separable and the numbers of clusters are a priori specified correctly then FCM correctly identifies the clusters. In fact, this is the case in every run of the FCM on these data sets, as can be seen from Figs. 4.3(a) and (b). However, as can be seen from Fig. 4.3(d) and Figs. 4.4(a) and (d), when the data are not linearly separable, FCM finds an artificial boundary based on the local density of the data distribution. FCM algorithm, as is well-known, clusters data in such a way that the hyper-volume of each of the clusters is roughly equal. It should be noted that in Fig. 4.3(c) and Figs. 4.4(b) and (c) though FCM appears to segregate the data well it is more due to the way the annular data are positioned, which makes it easy to linearly separate them.


(a) FCM on the data set (b) FCM on the data set in Fig. 4.1(a) in Fig. 4.1(b)

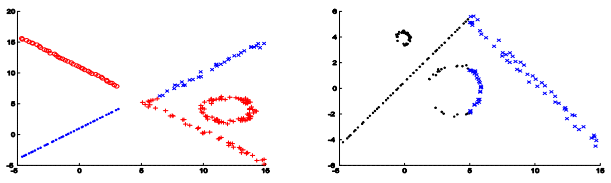

(c) FCM on the data set (d) FCM on the data set in Fig. 4.1(c) in Fig. 4.1(d)

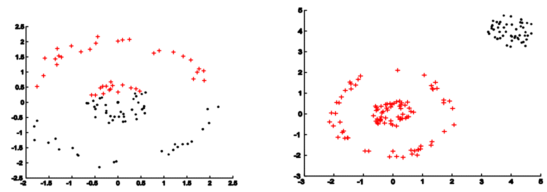
Figure 4.3: FCM on the data sets that contain both linear and Non-linear subsets.

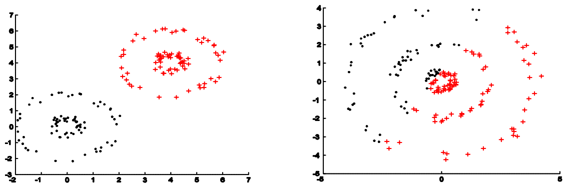
4.3 Gaussian Kernel-FCM on the data sets

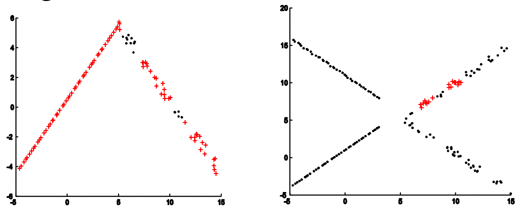
Figs. 4.5–4.7 contains the clusters that are usually found from repeated runs of the Gaussian Kernel-based FCM on these data sets. Note that the Gaussian kernel is given by the formula:

$$k(x,y) = \exp\left(\frac{-\lambda\|x-y\|^2}{2\sigma^2}\right) \tag{18}$$

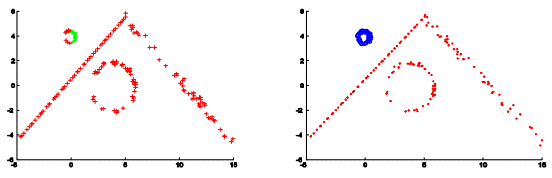
Unlike in the case of pure FCM, we have two degrees of freedom here in the form of the parameters λ, σ. It is found that when the data are only linearly separable, even if the numbers of clusters are a priori specified correctly for no value of the parameters λ, σ do we get a clear segregation. This can be seen from Figs. 4.5(a) and (b). Now let us consider the data sets in Figs.4.1(c) and (d), which contain linear and non- linear data sets. As can be seen from Figs. 4.6(a)–(d), the resulting clusters can vary significantly. The output obtained with λ = 2, σ = 0.5 are given

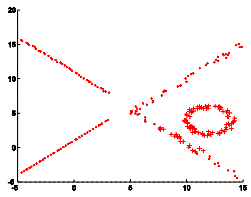

(a) FCM on the data set in Fig. 4.2(a) (b) FCM on the data set in Fig. 4.2(b)

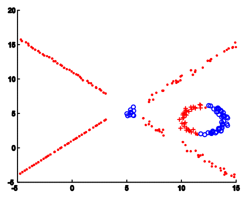

(c) FCM on the data set in Fig. 4.2(c) (d) FCM on the data set in Fig 4.2(d).
Figure 4.4: FCM on the non-linear data sets.


(a) GKFCM on the data set (b) GKFCM on the data set in Fig. 4.1(a) in Fig. 4.1(b)
Figure 4.5: Gaussian Kernel-based FCM on the linear data sets.

In Figs. 4.6(a) and (c), where we see a less than ideal segregation. However, when the parameters were set as follows λ = 0.5, σ = 0.5 with the number of clusters varying between c = 2, 3, 4. The best reasonable segregation we found on the data set in Fig.4.1(c) is given in Fig. 4.6(b) where we see that it has correctly clustered one of the circular data sets. Our results on the data set in Fig.4.1(d) with the above parameters did not show much improvement, see Fig. 4.6(d).


(a) GKFCM on the data set in (b) GKFCM on the data Fig. 4.1(c)with λ = 2, σ = 0.5 set in Fig. 4.1(c) with λ = 0.5, σ = 0.5

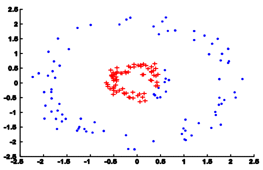


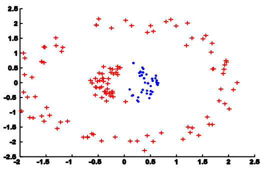


(c) GKFCM on the data set in Fig. 4.1(d) with λ = 2, σ = 0.5 (d) GKFCM on the data set in Fig. 4.1(d)with λ = 0.5 , σ = 0.5

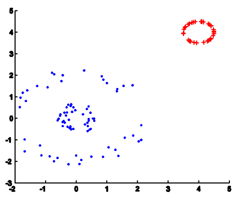
Figure 4.6: Gaussian Kernel-based FCM on the data sets that contain both Linear and Non-linear subsets

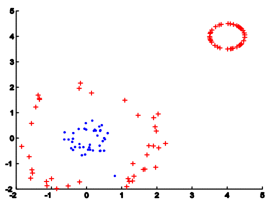
Finally, let us consider the data sets that are non-linear in $\mathbf{R}^2$. It can be seen that the annular ring data set which is usually touted as a shining example for the separation prowess of the Gaussian Kernel based FCM can also fail when the wrong parameters are chosen. Fig. 4.7(a) is with the correct choice of parameters, viz., λ = 0.5, σ = 0.5, while Fig. 4.7(b) shows the less than ideal separation produced with λ = 2, σ = 0.5. This is further emphasized by the results in Figs. 4.8(a) and (b) when applied on the data set in Fig. 4.2(d), where for no values of λ, σ can we segregate all the three annular rings. The Gaussian kernel-based FCM when applied to the data set in Fig. 4.2(b), with varying values of λ, σ and the number of clusters c = 3, always found the smaller of the circular data sets, but did not find the annular rings separately, see Figs. 4.7(c) and (d). It is interesting to note that when applied with c = 2 similar outputs were obtained.



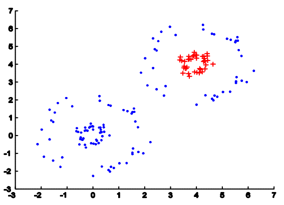


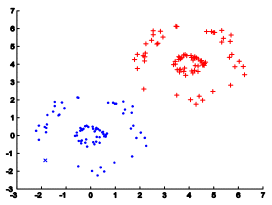
(a) GKFCM on the data set in Fig. 4.2(a) with λ = 0.5, σ = 0.5 (b) GKFCM on the data set in Fig. 4.2(a) with λ = 2, σ = 0.5





(c) GKFCM on the data set in Fig. 4.2(b) with λ = 0.5, σ = 0.5 (d) GKFCM on the data set in Fig. 4.2(b) with λ = 2, σ = 0.5



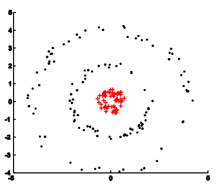


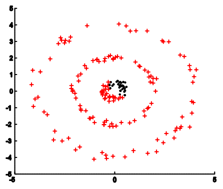
(e) GKFCM on the data set in Fig. 4.2(c) with λ = 0.5, σ = 0.5, c = 2 (f) GKFCM on the data set in Fig. 4.2(c) with λ = 0.5, σ = 0.5,c=4

Figure 4.7: Gaussian Kernel-based FCM on the non-linear data seta

Similarly, when applied to the data set in Fig. 4.2(c) with varying values of λ, σ and the number of clusters c the best results obtained were either of the following:

- Either, it found one of the inner annular rings data as one cluster and the rest of the data as a separate cluster, see Fig. 4.7(c), or
- It found both the annular rings to be separate clusters but did not identify the inner annular rings separately, see Fig. 4.7(d).





(a) GKFCM on the data set in Fig. 4.2(d) with λ = 0.5, σ = 0.5 (b) GKFCM on the data set in Fig. 4.2(d) with λ = 2, σ = 0.5

Figure 4.8: Gaussian Kernel-based FCM on the 3 Annular circles

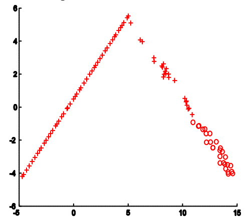
4.4 Tan-Hyperbolic Kernel-FCM on the data sets

Figs. 4.9–4.11 contain the clusters that are usually found from repeated runs of the Tan-Hyperbolic Kernel-based FCM on these data sets. Note that the Tan- Hyperbolic kernel is given by the formula:

$$k(x,y) = 1 - \tanh\left(\frac{\|x-y\|^2}{\sigma^2}\right) \quad (19)$$

Once again, we have a degree of freedom here in the form of the parameter σ. The results we obtained were very much along the lines of those obtained with the Gaussian kernel. It is found that when the data are linear, even if the number of clusters is a priori specified correctly for no value of the parameters σ do we get a clear segregation. This can be seen from Figs. 4.9(a) and (b).

Now let us consider the data sets in Figs.4.1(c) and (d), which contain both linear and non-linear data sets. As can be seen from Figs. 4.10(a) and (b), the resulting clusters can vary significantly. The output obtained with σ = 2 are given in Figs. 4.10(a) and (b), where we see a less than ideal segregation. While for the data set in Fig.4.1 (d) for no value of the parameters σ, c do we get a clear segregation, with σ=2 and the number of clusters varying between c = 2, 3, 4 the best reasonable segregation we found on the data set in Fig.4.1(c) is given in Fig. 4.10(a) where we see that it has correctly clustered one of the circular data sets.


(a) THKFCM on the data set in Fig. 4.1(a)

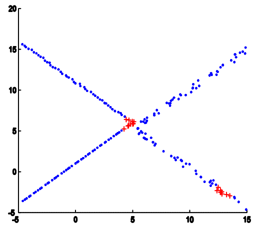
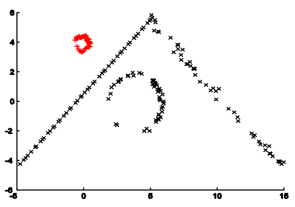

(b) THKFCM on the data set in Fig. 4.1(b)

Figure 4.9: Tan-Hyperbolic Kernel-based FCM on the linear data sets


(a) THKFCM on the data set in Fig. 4.1(c) with σ = 2

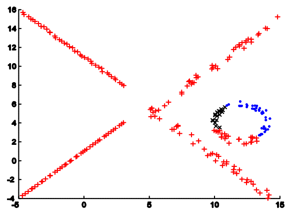
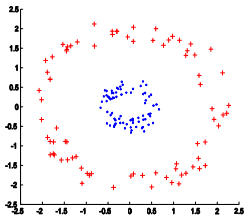

(b) THKFCM on the data set in Fig. 4.1(d) with σ = 2

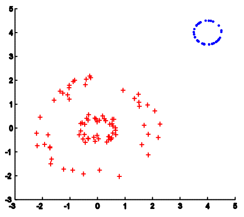
Figure 4.10: Tan-Hyperbolic Kernel-based FCM on the data sets that contain both Linear and Non-linear subsets.

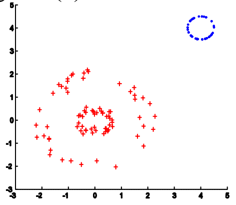
The Tan-Hyperbolic kernel FCM acts on the non-linear data sets in $\square^{2}$ much along the lines of the Gaussian Kernel FCM. It can be seen that with the right parameter chosen, in this case σ = 2 we obtain the correct result as given Fig. 4.11(a), while with other choices, say for instance with σ = 0.5 the clustering was very skewed as given in Fig. 4.11(e).

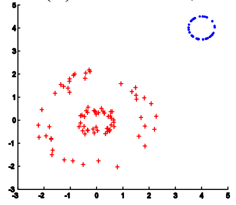
When applied to the data set in Fig. 4.2(b), with varying values of σ and the number of clusters c = 3, it usually found the smaller of the circular data sets, but did not find the annular rings separately, see Fig. 4.11(b). Once again, even when applied with c = 2 similar outputs were obtained.

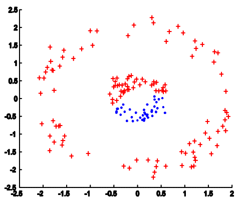
When applied to the data set in Fig. 4.2(c) and (d) with varying values of σ and the number of clusters c, the best results obtained mimicked those that were obtained with the Gaussian Kernel FCM (see Section 4.3), the results of which are:


(a) THKFCM on the data set in Fig. 4.2(a) with σ = 2


(b) THKFCM on the data set in Fig. 4.2(b) with σ = 2, c = 2


(c) THKFCM on the data set in Fig. 4.2(c) with σ = 2, c = 2


(d) THKFCM on the data set in Fig. 4.2(c) with σ = 2, c = 4


(e) THKFCM on the data set in Fig. 4.2(a) with σ = 0.5

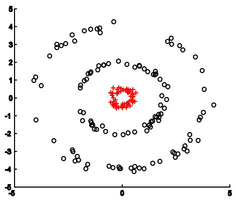

(f) THKFCM on the data set in Fig. 4.2(d) with σ = 2

Figure 4.11: Tan-Hyperbolic Kernel-based FCM on the non-linear data sets given in Figs. 4.11(c), (d) and (e).

5. Some Observations and Analysis based on our Results

5.1 Classical FCM

- The classical version of the FCM still remains the best suitable among the FCM stable when the data sets are linear, see Figs. 4.3(a) and (b).
- Classical FCM also has the advantage that except the fuzzifier m there are no parameters to choose. The conventional choice of m = 2 seems to work well in all our cases.
- The classical FCM performs poorly if the data are not linearly separable, see Figs. 4.3(c) and (d) and 4.4.

5.2 Kernel-based FCM

- The kernel-based FCMs tend to perform better than the FCM when the data sets are non-linear, see Figs. 4.7, 4.8 and 4.11.
- They offer many degrees of freedom in terms of their parameters.
- Setting these parameters correctly is a tough task and can affect the final result adversely as is shown in Figs. 4.7 and 4.11.
- In some cases, it may not be possible to find the right parameter even when the non-linearities in the data are known a priori.
- Their performance is poor when the underlying data sets are linear, see Figs. 4.9(a) and 4.9(b).
- Often they are not able to identify clusters of similar type, shape or size as is obvious from Figs. 4.6 and 4.10.

6. Conclusion

Based on our results, it appears that while the kernel-based modifications of FCM do possess many properties, their utility seems to be restricted to some specific type or native clusters and even here not all of those native clusters are identified and segregated well. By native clusters, we mean, for example spherical clusters of a particular radii when the Gaussian kernel with fixed λ, σ values are used. This is quite an interesting observation since in the kernelized FCM algorithm the kernel function essentially searches for the native clusters locally and should be able to identify all of these. This also explains the results obtained in Figs. 4.8 and 4.11. Further, there is no clear guidance towards the selection of the kernel function and a reference, even heuristic, with respect to which the optimization of the kernel parameters can be done.

In the case when the data contain both linear and non-linear subsets none of the versions of the FCM is able to identify all of them correctly. It is well-known that the fuzzifier m plays a major role in both the amount of overlap between clusters and also in removing unwanted local minima. In this case the effect of m was not considered and this merits a deeper exploration. In fact, it appears that even the kernelized versions of FCM tend to ensure that more or less all the clusters have the same hyper-volume, as in the case of FCM but perhaps in the feature space to where they are mapped. This also needs further exploration.

From the results obtained and for the stated reasons, it appears that the full potential of the kernelized versions of FCM are yet to be harvested and a far deeper exploration is necessary to lay thread bare clearly the context in which they are applicable with reasonable results when not much can be assumed on the data, especially when the data contain both linear and non-linear subsets.

References

[1] Mark Girolami, Mercer Kernel Based Clustering in Feature Space. IEEE Trans on Neural Networks, 13(3), 2002, pp.780-784.
[2] Sadaaki Miyamoto, Hidetomo Ichihashi, Katsuhiro Honda, Algorithms for Fuzzy Clustering, Springer-Verlag Berlin Heidelberg 2008.pp 9-29.
[3] B. Schoelkopf, A. J. Smola. Learning with Kernels-Support Vector Machines, Regularization, Optimization and Beyond, The MIT Press Cambridge, Massachusetts London, England 2002.
[4] R. Courant and D. Hilbert. Methooden der mathematischen Physik.Springer-Verlag, Berlin, 4th edition, 1993.
[5] D. Hilbert. Grundzuge einer allgemeinen Theorie der linearen Integral- gleichungen. Nachrichten der Gottinger Akademite der Wissenschaften, Mathematisch-Physikalische Klasse, pages 49-91, 1904.
[6] H. Konig, Eigenvalue Distribution of Compact Operators. Birkhauser, Basel, 1986.
[7] J. Mercer, Functions of positive and negative type and their connection with the theory of integral equations. Philosophical Transactions of Royal society, London, A 209: 415-446, 1909.

Author Profile

Man Singh received the M.Tech (Industrial Mathematics and Scientific Computing) from IIT Madras, Chennai, in 2012.